deep se
dependable evolvable pervasive software engineering group

Functional C++

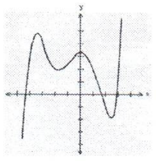
Federica Filippini - Marco Lattuada - Eugenio Gianniti

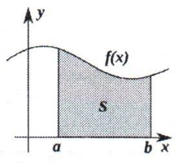
Politecnico di Milano
federica.filippini@polimi.it
marco.lattuada@polimi.it
eugenio.gianniti@polimi.it

POLITECNICO DI MILANO

Functions as Values

- Several algorithms should logically receive functions as parameters, they are higher order functions
  - Optimization techniques
  - Equation solvers
  - Numerical integration and differentiation





Using Callable Objects

- A way to work around this limitation is defining **callable objects**
  - Any class that implements a public `operator()` is callable

- `<functional>` helps in declaring **generic higher order functions**. How?

```
std::function <return_type (param1, param2, …)>
```

Example:

```
std::function <bool (double, double)>
```

can represent any function that compares two `double`s.

DEMO

Reference

- Lippman Chapters 10 & 14

- http://en.cppreference.com/w/cpp/header/functional